METHOD OF EMULATING A SHIFT REGISTER USING A RAM

FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to digital computation and, more particularly, to a method of emulating a very long shift register, using a random access memory (RAM) and a short shift register.

There are many applications in which a stream of incoming bits must be processed in real time. For example, it may be desired to apply a finite impulse response filter, each of whose coefficients is a single bit, to a stream of incoming bits. Denoting the K coefficients of the filter as $\{C_k\}$ ($k = 0$ through K-1) and the bits of the input stream (of indefinite length) as $\{X_n\}$, this means that the output of the filter is a set of numbers $\{Y_n\}$ such that

$$Y_n = \sum_{k=0}^{K-1} C_k X_{n-k}$$

and the indicated operation is an XOR operation.

The straightforward way to implement this filter would be to provide a register of length K to store the coefficients $\{C_k\}$ and a shift register of length K bits to store sequences of input bits $\{X_n\}$. As each new input bit arrives, the contents of the shift register are shifted over one bit to accommodate the new input bit. Note that this automatically discards the old input bit that preceded the new input bit by K bits. In-between arrivals of new input bits, an inner product operation is performed on the contents of the coefficient register and the shift register, to obtain the latest filter output.

This straightforward implementation suffers from the drawback that for filters of a useful length (for example, K=1024), a correspondingly long shift register is prohibitively expensive to fabricate on a processor chip. There is thus a widely recognized need for, and it would be highly advantageous to have, a method of emulating a shift register using a less expensive form of memory, such as RAM.

SUMMARY OF THE INVENTION

According to the present invention there is provided a method of processing successive input bits, including the steps of: (a) providing: (i) a RAM having a plurality of registers, each of the registers storing a word, all of the words being of equal length, (ii) a shift register at least as long as any of the words, and (iii) a pointer; (b) initializing the pointer to point to one of the registers of the RAM; and (c) for each group of j input bits: (i) writing the word, stored in the register pointed to by the pointer, to the shift register, (ii) shifting the word in the shift register by j bits, (iii) writing the group of j input bits to the shift register, thereby producing an updated word in the shift register, (iv) storing the updated word in the register pointed to by the pointer, and (v) incrementing the pointer.

A RAM typically consists of a group of individually addressable registers, each with a unique address, and in each of which a word of a certain length (typically 8, 16, 32 or 64 bits) may be stored and subsequently retrieved. The key to the present invention is the use of such a *word*-addressable memory to efficiently store successive individual input *bits* as they arrive. This is accomplished by also providing a relatively short (one word long) shift register and a pointer that encodes the addresses

of the RAM registers. The pointer is initialized to point to one of the RAM registers. As each input bit arrives, the word stored in the RAM register pointed to by the pointer is written to the shift register, shifted over one bit to make room for the new input bit, and written back to the RAM register whence it was retrieved. The pointer then is incremented to point to the next RAM register. Note that "incrementing" the pointer is defined herein cyclically: incrementing a pointer that points to the last RAM register produces a pointer that points to the first RAM register. Note that the input bits are stored in the RAM in transposed order, as explained more fully below.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

FIG. 1 is an illustration of a RAM at the beginning of an incoming bit storage cycle;

FIGS. 2A and 2B are illustrations of an N-bit shift register at two different stages of an incoming bit storage cycle;

FIG. 3 is an illustration of a RAM at the end of an incoming bit storage cycle.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention is of a method of emulating a very long shift register, using a RAM. Specifically, the present invention can be used to apply a finite-impulse-response filter to a sequence of bits, and to transpose a sequence of bits.

The principles and operation of shift register simulation according to the present invention may be better understood with reference to the drawings and the accompanying description.

Referring now to the drawings, Figure 1 shows a RAM **10** with M RAM registers, indexed $\boldsymbol{R_0}$ through $\boldsymbol{R_{M-1}}$, each capable of storing N bits, for a total capacity of NM bits, at a moment in time when NM bits X_n through X_{n-NM+1} of an incoming bit stream have been stored according to the present invention. The incoming bit which arrived earliest, X_{n-NM+1}, is stored in the (N-1)-th position in RAM register $\boldsymbol{R_0}$, the incoming bit which arrived next, X_{n-NM+2}, is stored in the (N-1)-th position in RAM register $\boldsymbol{R_1}$, and so on. The most recently arrived bit, X_n, is stored in the 0-th position in RAM register $\boldsymbol{R_{M-1}}$. A pointer **P** points to the RAM register, $\boldsymbol{R_0}$, that holds the earliest arriving stored incoming bit, X_{n-NM+1}, in its (N-1)-th position.

The arrival of the next incoming bit, X_{n+1}, initiates the next incoming bit storage cycle. The first step is to write the contents of the RAM register, $\boldsymbol{R_0}$, pointed to by pointer **P**, to an N-bit shift register **20**. Figure 2A shows N-bit shift register **20** at the end of this step. The second step is to shift the bits in N-bit shift register **20** up one position, discarding bit X_{n-NM-1} and making room in the 0-th position of N-bit shift register **20** for new incoming bit X_{n+1}. The third step is to store new incoming bit X_{n+1} in the 0-th position of N-bit shift register **20**. Figure 2B shows N-bit shift register **20** at the end of this step. The fourth step is to write the contents of N-bit shift register **20** to the RAM register, $\boldsymbol{R_0}$, pointed to by pointer **P**. Finally, pointer **P** is incremented to point to RAM register $\boldsymbol{R_1}$, which now is the RAM register which now holds the

earliest arriving stored incoming bit, X_{n-NM+2}, in its (N-1)-th location. Figure 3 shows RAM **10** at the end of this step.

In-between incoming bit storage cycles, the contents of RAM **10** may be read and manipulated in the conventional manner. For example, to produce the next output, Y_{n+1}, of the finite input response filter discussed above (assuming that $K=NM$), the M words stored in RAM **10** are successively read and XOR-ed with the coefficients $\{C_k\}$, which also are stored in M words in a different memory unit. Note that for this to be effected correctly, the coefficients $\{C_k\}$ must be stored in transposed order: C_{K-1}, C_{K-M-1}, C_{K-2M-1}, ... C_{2M-1}, C_{M-1}, C_{K-2}, C_{K-M-2}, C_{K-2M-2}, ... C_{2M-2}, C_{M-2}, ... C_{K-M+1}, C_{K-2M+1}, C_{K-3M+1}, ... C_{M+1}, C_1, C_{K-M}, C_{K-2M}, C_{K-3M}, ... C_M, C_0. This is precisely the transposed order produced by applying the method of the present invention to the bits $\{C_k\}$, treating the bits $\{C_k\}$ as an incoming bit string.

Typical values of M and N are 32 and 32, respectively.

Pointer **P** is incremented cyclically. Thus, in the incoming bit storage cycle in which pointer **P** initially points to the highest-indexed register, $\boldsymbol{R}_{M-1}$, "incrementing" pointer **P** means changing the value of pointer **P** to point to the lowest-indexed register, $\boldsymbol{R}_0$.

It will be appreciated that the principles of the present invention also are applicable to the processing of a stream of incoming bits other than one bit at a time. For example, the incoming bit stream may be processed three bits at a time, using RAM registers whose lengths are a multiple of 3 bits, and shifting the contents of the shift register by three bits in every incoming 3-bit storage cycle. The shift register must be at least as long as the RAM registers; if the shift register is used only for

unloading and loading the RAM registers, and the output of the shift register is not used in any other processing, then the length of the shift register need not be a multiple of 3 bits. The only practical limitation is that if bits are processed in groups of j, where j is a typical word length of a conventional RAM (for example, 8, 16 or 32), then the processing may as well be done word-wise rather than bit-wise, as described, for example, in U. S. Patent No. 5,568,443, to Dixon et al., in the context of the prior art.

While the invention has been described with respect to a limited number of embodiments, it will be appreciated that many variations, modifications and other applications of the invention may be made.